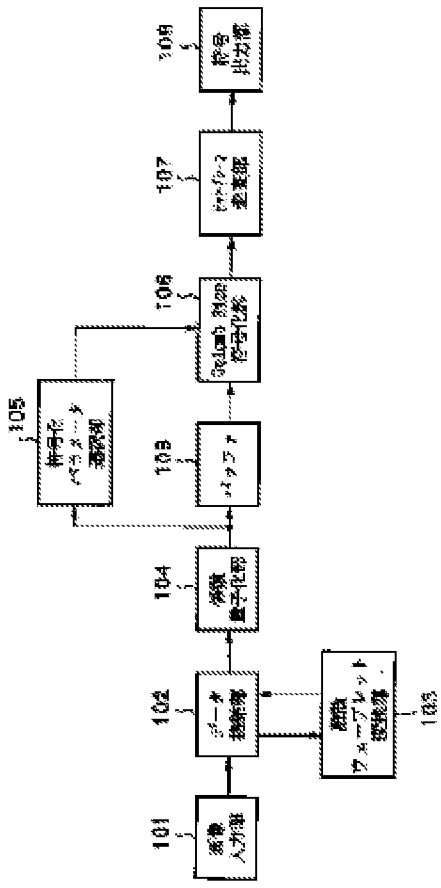
CODER, METHOD THEREFOR AND STORAGE MEDIUM

Publication number: JP2001044848
Publication date: 2001-02-16
Inventor: NAKAYAMA TADAYOSHI
Applicant: CANON KK
Classification:
- international: ***H04N1/41; H03M7/40; H04N1/413; H04N7/24; H04N7/26;*** **H04N1/41; H03M7/40; H04N1/413; H04N7/24; H04N7/26;** (IPC1-7): H03M7/40; H04N1/41; H04N1/413; H04N7/24
- European:
Application number: JP19990212713 19990727
Priority number(s): JP19990212713 19990727

Abstract of **JP2001044848**

PROBLEM TO BE SOLVED: To simply apply variable length coding to a multi-valued image at a high speed. SOLUTION: A data storage section 102 stores multi-valued image data received from an image input section 101, and a discrete wavelet conversion section 103 applies conversion processing to the stored data. As a result, data of sub blocks by each frequency component are fed to a coefficient quantization section 104, quantized according to quantization coefficient set by each sub block, and the quantized data are stored in a buffer 108. A coding parameter selection section 105 decides the optimum shift parameter for each quantized sub block and notifies a Golomb Rice coding section 160 of this.



Family list
5 family members for: **JP2001044848**
Derived from 5 applications

Back to JP2001044

1 **CODER, METHOD THEREFOR AND STORAGE MEDIUM**
Inventor: NAKAYAMA TADAYOSHI
Applicant: CANON KK
EC:
IPC: ***H04N1/41; H03M7/40; H04N1/413*** (+11)
Publication info: **JP2001044848 A** - 2001-02-16

2 **ENCODER**
Inventor: NAKAYAMA TADAYOSHI
Applicant: CANON KK
EC:
IPC: ***H04N7/26; H03M7/40; H04N1/41*** (+8)
Publication info: **JP2001077699 A** - 2001-03-23

3 **ENCODER AND ENCODING METHOD**
Inventor: NAKAYAMA TADAYOSHI
Applicant: CANON KK
EC:
IPC: ***G06F5/00; H03M7/30; H03M7/40*** (+12)
Publication info: **JP2001298623 A** - 2001-10-26

4 **CODER AND ITS METHOD**
Inventor: NAKAYAMA TADAYOSHI
Applicant: CANON KK
EC:
IPC: ***H04N7/30; H03M7/30; H03M7/40*** (+8)
Publication info: **JP2001298739 A** - 2001-10-26

5 **Coding apparatus and method**
Inventor: NAKAYAMA TADAYOSHI (JP)
Applicant: CANON KK (JP)
EC: H04N7/26L; H04N7/26A4V; (+4)
IPC: ***G06K9/46; H04N7/26***; **G06K9/46** (+2)
Publication info: **US6865299 B1** - 2005-03-08

Data supplied from the ***esp@cenet*** database - Worldwide

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号
特開2001－44848
（P2001－44848A）
(43)公開日　平成13年２月16日(2001.2.16)

(51)Int.Cl.⁷		識別記号	FI			テーマコード（参考）
Ｈ０３Ｍ	7/40		Ｈ０３Ｍ	7/40		５Ｃ０５９
Ｈ０４Ｎ	1/41		Ｈ０４Ｎ	1/41	Ｂ	５Ｃ０７８
	1/413			1/413	Ｚ	５Ｊ０６４
	7/24			7/13	Ｚ	９Ａ００１

審査請求　未請求　請求項の数15　ＯＬ　（全 17 頁）

(21)出願番号　特願平11－212713

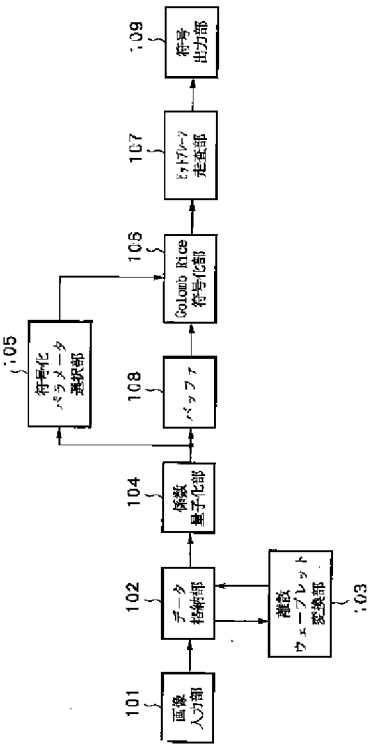
(22)出願日　平成11年７月27日(1999.7.27)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　中山　忠義
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内
(74)代理人　100076428
弁理士　大塚　康徳　（外２名）

最終頁に続く

(54)【発明の名称】　符号化装置及び方法及び記憶媒体

(57)【要約】
【課題】　多値画像を簡易かつ高速に可変長符号化することのできるようになる。
【解決手段】　画像入力部１０１より入力した多値画像データはデータ格納部１０２に格納され、離散ウェーブレット変換部１０３により変換処理が施される。この結果、各周波数成分毎のサブブロックのデータが係数量子化部１０４に供給され、各サブブロック毎に設定された量子化係数に従って量子化され、バッファー１０８に格納される。符号化パラメータ選択部１０５は、量子化された各サブブロック毎に、最適なシフトパラメータを決定し、それをＧｏｌｏｍｂ　Ｒｉｃｅ符号化部１６０に通知する。



【特許請求の範囲】
【請求項１】　あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を評価することにより、前記パラメータの最適値を決定することを特徴とする符号化方法。
【請求項２】　前記可変長符号部分の符号量の差分値は、パラメータの値が１だけ異なる場合、あるいは、２以上異なる場合の差分値であることを特徴とする前記請求項１記載の符号化方法。
【請求項３】　前記可変長符号部分の符号量の差分値を計算する時に、符号化するシンボル群の各ビットプレーン中の１の個数を表わす情報を用いることを特徴とする前記請求項１記載の符号化方法。
【請求項４】　前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列を、端から評価することを特徴とする前記請求項１記載の符号化方法。
【請求項５】　前記可変長符号部分の符号量の差分値を、パラメータ値の昇順あるいは降順に並べた差分値列の途中の値を評価した結果に基づき、それ以降の評価順序を決定することを特徴とする前記請求項１記載の符号化方法。
【請求項６】　あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量と、該符号量に所定の値を加算した値との比較結果に基づいて、前記パラメータの最適値を決定することを特徴とする符号化方法。
【請求項７】　あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化装置において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を計算する手段と、該差分値を所定値と比較する手段と、該比較結果に基づいて前記パラメータの最適値を決定する手段とを有することを特徴とする符号化装置。
【請求項８】　前記可変長符号部分の符号量の差分値は、パラメータの値が１だけ異なる場合、あるいは、２以上異なる場合の差分値であることを特徴とする請求項第７項記載の符号化装置。
【請求項９】　前記可変長符号部分の符号量の差分値を計算する手段は、符号化するシンボル群の各ビットプレーン中の１の個数を計数する手段を併せ持つことを特徴とする請求項第７項記載の符号化装置。
【請求項１０】　前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列を、端から所定値と順次比較する手段を有することを特徴とする請求項第７項記載の符号化装置。
【請求項１１】　前記可変長符号部分の符号量の差分値をパラメータ値の昇順あるいは降順に並べた差分値列の途中の値を所定値と比較する手段と、該比較結果に基づき、それ以降の比較順序を制御する手段と、該制御手段のもとで比較を行なう手段とを有することを特徴とする請求項第７項記載の符号化装置。
【請求項１２】　あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量を計算する手段と、該符号量に所定の値を加算する手段と、前記符号量と前記加算結果とを比較する手段と、該比較結果に基づいて、前記パラメータの最適値を決定する手段とを有することを特徴とする符号化装置。
【請求項１３】　二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化装置であって、
前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化する手段と、
量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定する手段とを備えることを特徴とする符号化装置。
【請求項１４】　二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化方法であって、
前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化し、
量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定することを特徴とする符号化方法。
【請求項１５】　コンピュータが読み込み実行することで、二次元画像データを離散ウェーブレット変換して量子化及び符号化する符号化装置として機能するプログラムコードを格納する記憶媒体であって、
前記離散ウェーブレット変換して得られた各周波数のサブブロック単位に、それぞれのサブブロック毎に設定された量子化係数に従って量子化し、
量子化された各サブブロックについて、シフトパラメータに従って固定符号部分と可変符号部分で符号化する際、当該シフトパラメータを、段階的に変化させながら前記可変長符号部分の差分値を評価することで決定するプログラムコードを格納する記憶媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は符号化装置及び方法

及び記憶媒体に関するものである。
【０００２】
【従来の技術】画像、特に多値画像は非常に多くの情報を含んでおり、その画像を蓄積・伝送する際にはデータ量が膨大になってしまうという問題がある。このため画像の蓄積・伝送に際しては、画像の持つ冗長性を取り除き、画質の劣化を多少許すことで、データ量の削減を図る高能率符号化が用いられている。
【０００３】これら高能率符号化の１つとしてＪＰＥＧ方式の圧縮符号化が知られている。この方式では多値画像をブロック毎にＤＣＴ変換することにより周波数成分に変換し、得られた変換係数を量子化し、可変長符号化するというものである。
【０００４】上記ＤＣＴ変換を用いた符号化は、圧縮率を高く設定すると復号画像にブロック歪みが発生してしまうという問題がある。近年、このような歪みを解消するためにウェーブレット変換を用いる新たな符号化方式が提案されている。
【０００５】また、これら種々の高能率符号化の一部として、可変長符号化が適用されることが多い。
【０００６】
【発明が解決しようとする課題】しかしながら、多値画像を簡単な処理で可変長符号化する為の方式、更にはこの可変長符号化方式を高速に実行するための処理方法については未だ確立されていない。
【０００７】本発明はかかる状況に鑑みてなされたものであり、多値画像を簡易かつ高速に可変長符号化することのできる符号化装置及び方法及び記憶媒体を提供することを目的とする。
【０００８】
【課題を解決するための手段】この課題を解決するため、例えば本発明の符号化装置は以下の構成を備える。すなわち、あるパラメータに基づいて、シンボル群を可変長符号部分と、パラメータの値が同じなら固定長となる符号部分とで符号化する符号化方法において、パラメータの各々の値に対する可変長符号部分の符号量の差分値を評価することにより、前記パラメータの最適値を決定する。
【０００９】また、本発明の好適な実施態様に従えば、多値画像データを複数の情報群（本実施の形態では１サブブロック分の量子化値の集まり）に分割し、該情報群を各々個別のパラメータ（ｋパラメータ）に基づき可変長符号化（Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化）する為の符号化装置であって、可変長符号部分の符号量が隣接するｋパラメータ間でどのくらい差があるのか、その差分値を求め、ｋパラメータの許容範囲内で求めた全ての差分値を、所定の値とを順次比較することにより、各情報群に最適なパラメータｋを決定する。
【００１０】
【発明の実施の形態】以下、添付図面に従って本発明に係る実施形態を詳細に説明する。
【００１１】［第１の実施形態］以下、８ビットのモノクロ画像データを符号化する例を第１の実施形態として説明するがあ、本発明はこれに限らず、各画素４ビットで表わすモノクロ画像、或いは各画素における色成分（ＲＧＢ／Ｌａｂ／ＹＣｒＣｂ）を８ビットで表現するカラーの多値画像を符号化する場合に適用することも可能である。これらに適用する場合には、各色成分をそれぞれモノクロ画像として符号化すればよい。
【００１２】図１は第１の実施形態における画像符号化装置のブロック構成図である。
【００１３】同図において、１０１は画像データ入力部、１０２はデータ格納部、１０３は離散ウェーブレット変換部、１０４は係数量子化部、１０５は符号化パラメータ選択部、１０６はＧｏｌｏｍｂ　Ｒｉｃｅ符号化部、１０７はビットプレーン走査部、１０８はバッファ、１０９は符号出力部である。
【００１４】まず、画像入力部１０１では符号化対象となる画素データをラスタースキャン順に入力する。この画像入力部１０１は、例えばスキャナ、デジタルカメラ等の撮像装置、或いはＣＣＤなどの撮像デバイス、或いはネットワーク回線のインターフェース等からの画像データ入力処理があげられる。
【００１５】また、画像入力対象はＲＡＭ，ＲＯＭ，ハードディスク、ＣＤ－ＲＯＭ等の記録媒体であってもよい。
【００１６】画像データ格納部１０２は画像入力部１０１で入力される画像データを、記憶する。
【００１７】離散ウェーブレット変換部１０３では、上記画像データ格納部に記憶した１画面分の画像データに対して公知の離散ウェーブレット変換を施し、複数の周波数帯域に分解する。本実施形態の形態では、画像データ列ｘ（ｎ）に対する離散ウェーブレット変換は次式によって行なうものとする。
【００１８】ｒ（ｎ）＝ｆｌｏｏｒ｛（ｘ（２ｎ）＋ｘ（２ｎ＋１））／２｝
ｄ（ｎ）＝ｘ（２ｎ＋２）－ｘ（２ｎ＋３）＋ｆｌｏｏｒ｛（－ｒ（ｎ）＋ｒ（ｎ＋２）＋２）／４｝
ｒ（ｎ），ｄ（ｎ）は変換係数であり、ｒ（ｎ）は低周波成分、ｄ（ｎ）は高周波成分である。
【００１９】また上式においてｆｌｏｏｒ｛Ｘ｝はＸを超えない最大の整数値を表わす。本変換式は一次元のデータに対するものであるが、この変換を水平方向、垂直方向の順に適用することで二次元の変換を行なうことが可能であり、図２（ａ）の様なＬＬ，ＨＬ，ＬＨ，ＨＨの４つの周波数帯域（サブブロック）に分割することができる。これらの変換データは、次段の係数量子化部１０４へスムーズにデータを出力したり、あるいは、更なるウェーブレット変換を行なうため、前記データ格納部１０２に記憶する。

【００２０】生成したＬＬ成分について前記同様の手順にて離散ウェーブレット変換を施すことにより図２（ｂ）のように７個の周波数帯域（サブブロック）に分解する。本実施形態の形態においては、更にもう一度繰り返して離散ウェーブレット変換を施すことにより図２（ｃ）に示すようにＬＬ，ＨＬ３，ＬＨ３，ＨＨ３，ＨＬ２，ＬＨ２，ＨＨ２，ＨＬ１，ＬＨ１，ＨＨ１の１０個の周波数帯域（サブブロック）に分割し、前記データ格納部１０２に記憶する。

【００２１】変換係数はＬＬ，ＨＬ３，ＬＨ３，ＨＨ３，ＨＬ２，ＬＨ２，ＨＨ２，ＨＬ１，ＬＨ１，ＨＨ１のサブブロック順に、かつ各サブブロック毎にラスタースキャン順にデータ格納部１０２から係数量子化部１０４へと出力する。

【００２２】係数量子化部１０４はウェーブレット変換係数の各々を各周波数成分毎に定めた量子化ステップで量子化し、量子化後の値を符号化パラメータ選択部１０５及びバッファ１０８へと出力する。

【００２３】係数値をＸ、この係数の属する周波数成分に対する量子化ステップの値をｑとするとき、量子化後の量子化値Ｑ（Ｘ）は次式によって求めるものとする。

【００２４】

Ｑ（Ｘ）＝ｆｌｏｏｒ｛（Ｘ／ｑ）＋０．５｝

本実施形態の形態における各周波数成分と量子化ステップとの関係を図３に示す。図に示すように低周波成分（ＬＬ等）よりも高周波成分（ＨＬ１，ＬＨ１，ＨＨ１）等の方が量子化ステップを大きくしている。これにより、視覚的に劣化の目立ちにくい高周波成分の情報の削減を行ない圧縮の向上を図っている。

【００２５】上記量子化値は、符号化パラメータ選択部１０５で評価され、後段のＧｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６でこの量子化値を符号化する際に必要となるｋパラメータが決定されるまで、バッファ１０８に蓄積保持される。

【００２６】符号化パラメータ選択部１０５では、上記量子化値に基づいて後段のＧｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６で使用されるｋパラメータを選択する。

【００２７】このｋパラメータとはＧｏｌｏｍｂ　Ｒｉｃｅ符号化という可変長符号化を行なう際の固定長部の符号長を示す値である。説明の都合上、符号化パラメータ選択部１０５の説明する前に、Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６の動作から先に説明する。

【００２８】このＧｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６で行なう符号化の基本的な方法は公知ではあるが、以下に符号化の基本的な動作及び本発明の特徴的な部分について説明する。

【００２９】まず、量子化値を格納したバッファ１０８から１サブブロック分のデータを順次読み出す。

【００３０】次に、Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６で、該サブブロックに対応するｋパラメータに基づき、以下に説明する前処理やＧｏｌｏｍｂ　Ｒｉｃｅ符号化処理を行なう。

【００３１】Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６は入力した各量子化値の正／負を調べ、負号ビットを出力する。具体的には量子化値が０または正である場合には「０」を、負である場合には「１」を出力した後、該量子化値を絶対値に変換する。（この処理は厳密にはＧｏｌｏｍｂ　Ｒｉｃｅ符号化処理には含まれないが、ここでは前処理として、Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６内で行なうものとする）

次に、該絶対値をＧｏｌｏｍｂ　Ｒｉｃｅ符号化する。符号化対象となる量子化値の絶対値をＶ、処理対象のサブブロックに適用されるｋパラメータの値がｋである場合のＧｏｌｏｍｂ　Ｒｉｃｅ符号化処理は次の手順にて行なう。

【００３２】まず、２進表現したＶをｋビット右シフトした後に、得られた値（残った値）の個数だけ「０」を連続して配置し、その次に区切りのビットとして「１」を配置し、その次に元のＶの下位ｋビットを配置することにより可変長符号（Ｇｏｌｏｍｂ　Ｒｉｃｅ符号）を生成する。

【００３３】図４にＶ＝０～７、ｋ＝０，１，２に対するＧｏｌｏｍｂ　Ｒｉｃｅ符号の例を示す。例えば、Ｖ＝４で、ｋ＝１の場合、４は二進数で“１００Ｂ（Ｂは二進数を示す）”であるから、ｋ（＝１）ビット右シフトすると、残りは“１０Ｂ”、つまり“２”であるから、“０”を２回続け、その後に区切りの“１”、そして値４（＝１００Ｂ）の下位ｋ（＝１）ビットにより、“００１０”を得る。

【００３４】これから分かるように、絶対値Ｖ及びパラメータｋから得られる各可変長符号の符号長は、Ｖ＞＞ｋ（Ｖをｋビット右シフトしたときに残った値）＋１（区切りビット）＋ｋ（パラメータの値）ビットであることが容易に推測できる。正／負を表わす負号ビットを加えればさらに１ビット増える。

【００３５】また、図からも分かるように、ｋパラメータ＝０の時には量子化値「０」に対するＧｏｌｏｍｂ　Ｒｉｃｅ符号長がｋ＝１，２の時よりも特に短くなっている。これは符号化される量子化値群が０に偏っているため、ｋパラメータを小さくして符号化する方が適していることを示している。

【００３６】また、ここで用いるＧｏｌｏｍｂ　Ｒｉｃｅ符号化は、図４に示すような符号表を実際には保持することなく簡単な演算により符号化及び復号化を行なうことができるという効果がある。通常、ハフマン符号化等の可変長符号化では、符号化対象値に対する可変長符号を示すテーブルを保持しなければならない。特に、関連する状態に応じ、符号化対象値に対して複数の可変長符号を適宜切り換える場合には、上記テーブルを複数持つ必要がある。

【００３７】以上のようにして、入力される量子化値に対する負号ビット（＋／－を表わす）と可変長符号（Ｇｏｌｏｍｂ　Ｒｉｃｅ符号）からなる符号化データを生成し、後段のビットプレーン走査部１０７へ出力する。

【００３８】次に、上述したＧｏｌｏｍｂ　Ｒｉｃｅ符号化部１０６で必要となるｋパラメータを生成する符号化パラメータ選択部１５０の処理内容の説明を行なう。

【００３９】符号化するサブブロックの全符号量Ｔは、ｋパラメータの値をｐとすると

$$T_p=\Sigma(Vi>>p\ (Viをpビット右シフトした値)+2\ (負号ビットと区切りビット)+p)\cdots\cdots(1)$$

となる。

【００４０】符号化するサブブロック内の量子化値の数をＮ個とすると、上記（１）式は下式のようになる。

【００４１】

$$T_p=(p+2)\times N+\Sigma Vi>>p\quad\cdots(2)$$

上記（２）式の第１項：（ｐ＋２）×Ｎは固定長符号部分による符号量で、第２項：ΣＶｉ＞＞ｐは可変長符号部分による符号量である。ここで第２項のみを

$$S_p=\Sigma Vi>>p\quad\cdots(3)$$

と表わす。このS_pはｐの値が小さい程大きくなる。一方、固定長符号部分の符号量はｐの値に比例して大きくなり、ｐが１つ大きくなると該符号量はＮビット増える。

【００４２】次に、Ｖｉを２進表現したときの各ビットプレーンにおける「１」の個数について考える。

【００４３】Ｖｉ＞＞ｐの最下位ビットのプレーン中に存在する「１」の数をB_{p+1}とする。すなわち、

$$B_{p+1}=\Sigma(Vi>>p)\&1\cdots\cdots(4)$$

である。ここで、B_1はＶｉの最下位ビットプレーン中の「１」の個数を表わし、B_2はＶｉの最下位ビットから２番目のプレーン中の「１」の個数を表わす。

【００４４】$\Sigma Vi>>p=(2\times\Sigma Vi>>(p+1))+(\Sigma(Vi>>p)\&1)$

という関係と、（３）（４）式から

$$S_p=2\times S_{p+1}+B_{p+1}\cdots\cdots(5)$$

という関係式が得られる。ｋパラメータの値が１つだけ異なるS_{p-1}とS_p間の差をD_pとすると、

$$D_p=S_{p-1}-S_p=(2\times S_p+B_p)-S_p$$
$$=S_p+B_p\ (p\geqq 1)\cdots\cdots(6)$$

このD_pはｋパラメータの値がｐからｐ－１に変わることで、可変長符号量がどれだけ増えるかを表わすもので、この値がＮより小さいなら、全体の符号量はＮ－D_pだけ減るので、ｋパラメータの値はｐよりｐ－１にした方がよい。逆に、$D_p\geqq N$ならｋパラメータの値はｐの方がよい。

【００４５】同様に、ｋパラメータの値が２つ異なるS_{p-2}とS_p間の差をDD_p、すなわち、$DD_p=S_{p-2}-S_p$とすると、この値が２Ｎより小さいなら、ｋパラメータの値がｐからｐ－２に変わることで可変長符号量が２Ｎ－DD_pだけ減るので、ｋパラメータの値はｐよりｐ－２にした方がよい。逆に、$DD_p\geqq 2N$ならｋパラメータの値はｐの方がよいと言える。

【００４６】上記（６）式をさらに変形すると、

$$D_{p-1}=S_{p-1}+B_{p-1}$$
$$=(2\times S_p+B_p)+B_{p-1}$$
$$=(S_p+B_p)+S_p+B_{p-1}$$
$$=D_p+S_p+B_{p-1}$$

S_p，$B_{p-1}\geqq 0$であるから

$$D_{p-1}\geqq D_p\cdots\cdots(7)$$

上記（７）式の関係と上述した説明から次のことが言える。

【００４７】『数列：D_1、D_2、D_3、・・、D_{p-1}、D_p、D_{p+1}、・・、D_mには、D1≧D2≧D3≧・・≧Dp-1≧Dp≧Dp+1≧・・≧Dmという関係があり、$D_q\geqq N\geqq D_{q+1}$となるｑが存在するとき、このｑはｋパラメータの最適値の内の１つである。』

ここでいう最適値とは、Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化したときの全体の符号量が最小になるという意味である。また、ｍはゼロでないB_pの中で最も大きなｐの値であり、このｍに対応するS_mはゼロある。（一番大きな量子化絶対値をｍビット右シフトすると０になるため）よって、$D_m=B_m\leqq N$である。

【００４８】ｋパラメータがｑからｑ＋１になると符号量がＮ－D_{q+1}だけ増加し、ｋパラメータがｑからｑ－１になると符号量がD_{q-N}だけ増加する。但し、$D_{q+1}=N$または$D_q=N$の時は符号量は増えず最小のままである。$D_{q+1}=N$の時はｑ＋１も最適値となり、$D_q=N$の時はｑ－１も最適値となる。

【００４９】最適値は最大３個存在しうる。その条件は$D_q=N=D_{q+1}$が成立する時で、言い替えると$S_{q-1}=0$、$S_q=N$、$S_{q+1}=2N$となる時である。この時、ｑ－１、ｑ、ｑ＋１が最適値となり、全符号量Ｔは極値（最小値）をとる。よって、ｋパラメータが上記最適値から遠ざかる程、全符号量は増加する。

【００５０】一方、上記ｑが存在しない時というのは、$N>D_1$となる時で、この場合ｋパラメータの値が０で全符号量が最小となる。そして、全符号量はｋの値の変化（増加）に対し単調増加となる。

【００５１】ここで、前記数列と最適値に関する関係を一般化するために、$D_0=D_1+N$と定義する。すると、『数列：D_0、D_1、D_2、D_3、・・、D_{p-1}、D_p、D_{p+1}、・・、D_mには、$D_0\geqq D_1\geqq D_2\geqq D_3\geqq\cdots\geqq D_{p-1}\geqq D_p\geqq D_{p+1}\geqq\cdots\geqq D_m$という関係があり、$D_q\geqq N\geqq D_{q+1}$となるｑが存在し、このｑはｋパラメータの最適値の内の１つである。』ということが言え、最適値ｑの存在が明確なものとなった。

【００５２】これまでの説明から、ｋパラメータの変化に対する全符号量Ｔｋの局所的な最小値は、ｋパラメータの全範囲におけるグローバルな最小値でもあることが分かる。よって、ｋパラメータの最適値の探索は、ｋの値すべてについて全符号量Ｔｋが最小になるかどうかを評価する必要はなく、局所的にＴｋが最小値となるｋの値を見つけ出すだけでよいと言える。これにより、ｋパラメータの探索方法の簡略化が図れる。

【００５３】以上で述べたさまざまな関係から、ｋパラメータの最適値を決定するいくつかの方法（発明の実施形態）が考えられるが、本第１の実施形態の例を図５のフローチャートに示す。

【００５４】まず、ステップＳ５０１では、可変長符号部分の符号量S_p（ｐ＝０、１、２、・・、ｍ－１）を計算し、ステップＳ５０１ａでは、ゼロでないS_pのｐの最大値に１を加算してｍを求める。

【００５５】ステップＳ５０２では、ｉに上記ｍを設定し、ステップ５０３では、$D_i=S_{i-1}-S_i$を計算する。

【００５６】ステップＳ５０４では、D_iをＮと比較し、$D_i<N$なら、ステップＳ５０５へ、そうでなければ、ステップＳ５０７へ行く。

【００５７】ステップ５０５では、ｉ＝ｉ－１を計算してｋパラメータの候補値を更新する。そして、ステップ５０６では、更新したｉの値が０かどうかを判定し、ｉ＝０であればステップ５０７へ行き、ｉ＝０でなければステップＳ５０３へ行ってループ処理を継続する。

【００５８】ステップＳ５０７では、ｋパラメータの値をｉに決定する。

【００５９】ステップＳ５０１における、可変長符号部分による符号量S_pの計算プログラム例（Ｃ言語で記述）を図６に示す。この例では、Ｖ＝０のときにも最低１回の加算を行なうが、ｉｆ文でＶ≠０を検出したときだけ、ｄｏ～Ｗｈｉｌｅ文を実行するようにしてもよい。

【００６０】サブブロック内の全て（Ｎ個）のＶに対して図６の処理を行なったら次のステップＳ５０１ａでｋパラメータの上限候補値ｍを求める。この値は、前記Ｖの理論上の最大値から図６における変数ｐの理論上の最大値をあらかじめ計算しておき、Ｓ［］の値を順に評価することで求められる。

【００６１】そして、ステップＳ５０２において、該ｍをｉに設定する。

【００６２】その次にステップＳ５０３にて$D_m=S_{m-1}-S_m$を計算する。このD_mは全てのD_iの中で一番小さく、最大でもＮであり、Ｎより大きくなることは有り得ない。これは最も大きな値Ｖの最上位の有意ビット（０で無いビット）のビットプレーンに「１」が存在する個数を表わしている。

【００６３】よって、ステップＳ５０４で、$D_{i(=m)}<$Ｎと判定され、ステップＳ５０５へ移る。

【００６４】ステップＳ５０５でｉの値はｍから１つ減らされｍ－１となり、次のステップＳ５０６でｉの値が０かどうか判定されるが、ここではｉ≠０としてステップＳ５０３へ戻る。

【００６５】上記、ステップＳ５０３～Ｓ５０６の処理を繰り返し行ない、$D_i>N$またはｉ＝０になるとｋパラメータの値を変更しても符号量が減らないため、ステップＳ５０７に抜けて、ｋパラメータの値をｉに決定する。

【００６６】以上のようにして、図１における符号化パラメータの選択部で処理を行ない、該パラメータに基づき、既に説明したＧｏｌｏｍｂ　Ｒｉｃｅ符号化処理部１０６を実行する。符号化されたデータは、次段のビットプレーン走査部１０７に出力される。

【００６７】ビットプレーン走査部１０７は、量子化値毎に符号化・出力された符号化データを受け取り、それをビットプレーン単位に並び替えて出力するものである。これを、上述した周波数成分（サブブロック）単位で行なう。

【００６８】まず、Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部で生成された符号化データをｋパラメータでの値で決まる固定長符号部分と可変長符号部分に分離する。符号にはその外に１ビット固定の負号ビットがある。

【００６９】走査する順序は、重要な情報からで、まず最初に負号ビットをサブブロック内の全量子化値に対して走査する。次に、量子化値の上位ビットの情報を有する可変長符号部分を上位ビットプレーンから順に走査する。最後に固定長符号部分を上位ビットプレーンから順に走査する。各量子化値に対するＧｏｌｏｍｂ　Ｒｉｃｅ符号（但し、可変長符号部分と固定長符号部分とを分離したもの）と走査順序を図７に示す。

【００７０】可変長符号部分の走査についてはもう少し詳しい説明を付け加える。可変長符号はその名の通り長さが一定でないため、短い符号と長い符号をどのようにそろえて並べるかということが一意に決まらないが、ここでは、可変長符号の先頭ビットが同じプレーンになるように並べるものとする。すなわち、可変長符号は図７における可変長符号領域の上から順にうめるようにして並べる。上記ビットプレーン情報を上位ビットから順に走査すると、短い符号では途中のプレーンから情報（符号ビット）が無くなるので情報の無い符号は走査をスキップする。よって、下位のビットプレーンへ行くにつれ、走査する情報（符号ビット）が少なくなる。

【００７１】上記順序（負号ビット、可変長符号、固定長符号）で走査したビット情報の並びを図８に示す。

【００７２】このビットプレーン走査をサブブロック単位でＬＬ，ＨＬ３，ＬＨ３，ＨＨ３，ＨＬ２，ＬＨ２，ＨＨ２，ＨＬ１，ＬＨ１，ＨＨ１の順に行なう。

【００７３】このビットプレーン走査部から出力される

ビット列を次の符号出力部１０９にて順次転送する。転送相手は、ハードディスクやＤＶＤ等の記録メディアであっても良いし、インターネット、一般公衆回線、無線回線等のインターフェースであっても良い。

【００７４】なお、上記実施の形態において生成された符号化データには、復号時に必要となる各種情報、すなわち、画像サイズ、１画素あたりのビット数、各周波数成分に対する量子化ステップ、ｋパラメータの値等を付属情報として符号化データに適宜付加する。

【００７５】［第２の実施形態］上記第１の実施の形態における符号化パラメータの選択方法、すなわち、ｋパラメータの生成方法とは若干異なる方法を図９に示し説明する。なお同じ役割を担う処理ステップには図５のものと同一の番号を付加している。

【００７６】本第２の実施形態は、前記（６）式：$D_p=S_p+B_p$を利用する。よって、ステップＳ５０１の替わりにステップＳ９０１、ステップＳ５０３の替わりにステップＳ９０３にて処理を行なう。

【００７７】ステップＳ９０１では、可変長符号部による符号量S_p（ｐ＝０、１、２、・・、ｍ－１）と各ビットプレーンにおける「１」の数B_p（ｐ＝１、２、・・、ｍ）を計算し、ステップＳ９０３では、$D_i=S_i+B_i$を計算する。

【００７８】ステップＳ９０１のS_pとB_pの計算プログラム例（Ｃ言語で記述）を図１０に示す。該プログラムにおけるｄｏ～Ｗｈｉｌｅループ文の第１巡目でS0、B1に対して累積加算処理を行ない、第ｐ巡目ではSp-1、Bpに対して累積加算処理を行なうようにしている。

【００７９】本第２の実施形態の最大の特徴は、D_iを前記（６）式で計算することにある。そのために必要なS_iとB_iをステップＳ９０１にてあらかじめ求めておき、ステップＳ９０３にてD_iを計算する。その他の処理は前記第１の実施形態と同じであるため、説明は省略する。

【００８０】［第３の実施形態］本第３の実施形態は、前記（５）式：$S_p=2\times S_{p+1}+B_{p+1}$の関係を利用する。すなわち、前記第２の実施形態におけるステップＳ９０１の処理で、S_pとB_p（ｐ＝０、１、２、・・、ｍ）を計算する部分を、以下に示すステップＳ９１１～ステップＳ９１３の処理に置き換える。

【００８１】ステップＳ９１１では、B_p（ｐ＝０、１、２、・・、ｍ）を計算する。

【００８２】ステップＳ９１２では、ゼロでないB_pのｐの最大値ｍを求める。

【００８３】ステップＳ９１３では、$S_m=0$に初期化し、$S_{p-1}=2\times S_p+B_p$（ｐ＝ｍ、ｍ－１、ｍ－２、・・、１）を計算する。

【００８４】その他の処理は、前記第２の実施形態と同じであるので、説明を省略する。なお、ステップＳ９１３の処理を「$S_{m-1}=B_m$に初期化し、$S_{p-1}=2\times S_p+B_p$（ｐ＝ｍ－１、ｍ－２、・・、１）を計算する。」と変更してもよい。

【００８５】［第４の実施形態］本第４の実施の形態は、前記（６）式の関係：$D_p=S_p+B_p$から容易に導き出せる　$D_p\leqq S_p$という関係を利用する。$D_p\leqq S_p$という関係より、$S_p<N$であれば、$D_p<N$であると言える。

【００８６】そこで、図１１に示すフローチャートのように、$S_i<N$の関係が成り立つかどうかを調べるステップＳ１１０１を前記第２の実施の形態を示す図９のステップＳ９０３の前に設け、該関係が成り立てば、ステップＳ９０３とステップＳ５０４をパスするようにした。

【００８７】なお、本実施形態は、前記第１の実施形態に適用することも可能である。その場合、ステップＳ１１０１を図５のステップＳ５０３の前に設け、該関係が成り立てば、ステップＳ５０３とステップＳ５０４をパスするようにする。

【００８８】［第５の実施形態］上記第１、第２の実施の形態における符号化パラメータの選択方法とは異なる、更に別の方法について説明する。

【００８９】本実施の形態では、ｋパラメータの値が２つ異なるSp-2とSp間の差、すなわち、$DD_p=S_{p-2}-S_p$を利用する。

【００９０】既に説明したように、DD_pが２Ｎより小さいなら、ｋパラメータの値をｐからｐ－２に変えることで可変長符号量が$2N-DD_p$だけ減るので、ｋパラメータの値はｐよりｐ－２にした方がよい。逆に、$DD_p\geqq 2N$ならｋパラメータの値はｐの方がよいと言える。

【００９１】よって、該DD_pを用いれば、ｋパラメータの候補値を２単位で高速に更新できる。本実施形態の処理方法を示すフローチャートを図１２に示す。同じ役割を担う処理ステップには図５のものと同一の番号を付加している。

【００９２】本実施形態では、前記第１の実施形態における図５のステップＳ５０２とステップＳ５０３の間に別のループ処理を行なうステップＳ１２０１～ステップＳ１００４を設けている。

【００９３】ステップＳ１２０１では、本実施形態の一番の特徴である$DD_i=S_{i-2}-S_i$を計算する。ステップＳ１２０２では、DD_iと２Ｎとを比較する。ステップＳ１２０３では、ｉを２単位で更新する。ステップＳ１２０４では、ｉが１以下であるかどうかを判定する。ステップＳ１２０１～ステップＳ１２０４によるループ処理は、前記第１の実施形態におけるステップＳ５０３～ステップＳ５０６によるループ処理と大変よく似ている。

【００９４】違いはｉを更新する単位が２であるということである。ステップＳ１２０１で計算したDD_iをス

テップＳ１２０２で２Ｎと比較することにより、ｉを２つ更新してもよいかどうか判定し、更新可能であれば、ステップＳ１２０３にてｉ＝ｉ－２によりｉの値を更新する。

【００９５】ｉ≦１ではステップＳ１２０１を実行できないので、（何故ならS_{-1}が存在しないため）ステップＳ１２０４にて、ｉ≦１を検出したら次のステージの処理（ステップＳ５０３）へ移り、ｉ＞１であれば、ステップＳ１２０１へ戻ってループ処理を継続する。

【００９６】上記ループ処理を抜けてきた時というのは、ｋパラメータをｉからｉ－２にしても符号量が減らないという状態である。しかし、ｋパラメータをｉからｉ－１にすると符号量が減る可能性がある。

【００９７】そこで、ｋパラメータをｉからｉ－１に変更可能かどうかを調べるため、ステップＳ５０３で$D_i=S_{i-1}-S_i$を計算し、ステップＳ５０４でD_iをＮと比較し、D_i＜Ｎなら、ｉからｉ－１に変更できるので、ステップＳ５０５に移り、そうでなければ、それまでのｉをそのまま引き継いでステップＳ５０７へ行き該ｉの値をｋパラメータの値とする。

【００９８】本実施形態では、まず最初にｉを２単位で更新したが、$S_{i-4}-S_i$を用いれば、４単位で更新することもできるし、他の計算式を用いれば別の単位で更新することも可能である。

【００９９】４単位で更新した場合、最初のループ処理を抜けた時点で、最適なｋパラメータの可能性としては、ｉ～ｉ－３の４つが考えられる。そこで、次のステージで、この４つの値のどれかに特定するため、１単位で更新可能かどうかをループ処理で最大３回判定し、ｉの値を更新する、といった処理方法も考えられる。

【０１００】［第６の実施形態］符号化パラメータの選択方法として、更に異なる方法を図１３に示し説明する。なお同じ役割を担う処理ステップには図５のものと同一の番号を付加している。

【０１０１】本実施形態は、D_iを使用せずに、$SN_p=S_p+N$（ｐ＝０、１、２、３、・・、ｍ）という値を用いることが特徴である。

【０１０２】ステップＳ１３０１では、上記SN_pを計算し、ステップＳ１３０２では、S_{i-1}とSN_iを比較する。

【０１０３】その他の処理は前記第１の実施の形態と同じである。

【０１０４】ステップＳ５０１で計算したS_p（ｐ＝０、１、２、３、・・、ｍ）の値に対して、Ｎを加算した値SN_pをステップＳ１３０１で計算した後、ステップＳ５０２でｉの初期化を行なう。

【０１０５】その後、ｋパラメータの候補値であるｉをｉ－１へ更新可能かどうかを調べるため、ステップＳ１３０２にて、S_{i-1}とSN_iを比較する。$S_{i-1}<SN_i$であれば符号量が減るので、ｉをｉ－１へ更新するためステップＳ５０６へ行き、そうでなければ、ステップＳ５０４に抜けｋパラメータの値をｉに確定する。

【０１０６】ｉをｉ－１へ変更した後、ステップＳ５０６にて、ｉ＝０かどうか判定する。ｉ＝０になるとｉの値をそれ以上更新できないので、ループ処理を抜けステップ５０４へ行く。そうでなければ、ステップＳ１３０２へ戻り、ｉの更新処理を継続する。

【０１０７】［第７の実施形態］これまで説明した実施形態は、すべてループ制御変数ｉの初期値をｍとしてきたが、本実施の形態では、該制御変数ｉの初期値を０に設定するものである。

【０１０８】基本的には、これまで述べてきた全ての実施形態に適用可能であるが、特に第１の実施形態である図５を基にして、本実施の形態を説明する。ｉの初期値を０とする本実施形態を図１４に示す。

【０１０９】同図において、ステップＳ１４０２では、ｉに初期値０を設定し、ステップＳ１４０４では、D_iをＮと比較し、D_i＞Ｎならステップ Ｓ１４０５へ、そうでなければステップＳ１４０７へ行く。ステップＳ１４０５では、ｉに１を加算し、ステップＳ５０３に戻る。

【０１１０】ステップＳ１４０７では、ｋパラメータの値をｉ－１に決定する。

【０１１１】本実施形態では、これまでの実施形態で必要であった、ｍを求めるためのステップＳ５０１ａ、ｉが０かどうかを判定するステップＳ５０６が必要なくなった。

【０１１２】以前述べたように$D_m\leqq N$であるため、ｉの値がｍまで大きくなるか、それ以前に必ず、$D_i\leqq N$という関係になるため、ステップＳ１４０４の分岐先は最後は必ずステップＳ１４０７になり、ここでｋの値が確定する。

【０１１３】ステップＳ１４０７でｋパラメータの値を、ｉではなくｉ－１に設定する理由は、ステップＳ１４０４からステップＳ１４０７に分岐した時点での諸関係が、$D_{i-1}>N\geqq D_i$であるからである。前述したように、該関係でのｋパラメータの最適値はｉ－１である。

【０１１４】本実施形態は、これまでの中で最もシンプルな処理形態をとることができる。

【０１１５】［第８の実施形態］本実施形態は、前記制御変数ｉの初期値を、それ以前に行なう比較処理の結果によって変えるというものである。

【０１１６】ｋパラメータの最適値は、ｉ＝ｍに近いところに存在する場合もあれば、ｉ＝０に近いところに存在する場合もある。前者であればｉ＝ｍから、後者であればｉ＝０から探索するのが合理的というものである。

【０１１７】ループ処理に入る前にまずその判定をしなければならない。ｈ＝ｆｌｏｏｒ｛ｍ／２｝に対する、D_hを計算し、該D_hとＮとを比較する。D_h＜Ｎであれば、ｉを０に設定して処理を行ない、そうでなければ、

ｉをｍに設定して処理を行なう。というのが本実施形態の概略である。

【０１１８】本実施形態を図１５に示し、もう少し詳しく説明する。同図において、ステップＳ１５０２では、ｈ＝ｆｌｏｏｒ｛ｍ／２｝を計算する。ステップＳ１５０３では、上記ｈに対応するD_hを計算する。

【０１１９】ステップ１５０４では、D_hとＮとを比較し、D_h＜Ｎであればステップ１４０２へ分岐し、そうでなければ、ステップＳ５０２へ分岐する。

【０１２０】その他の処理ステップは図５あるいは図１４の同一番号処理ステップと同じである。

【０１２１】ステップＳ１４０２以降の処理は、前記第７の実施形態とまったく同じ処理であるが、ステップＳ５０２以降の処理は、基本的には前記第１の実施形態と同じであるが、若干違いがある。その違いは、ループ処理において、ｉが所定の値に達したかどうかの判定（ステップＳ５０６に相当する）が無いことである。

【０１２２】何故判定が無いかというと、該ループ処理においてｉの値が初期値ｍから１ずつ減少して上記ｈに達すると、ステップＳ５０４の判定で該ループ処理を抜けステップＳ５０７へ確実に分岐することが分かっているからである。これはステップＳ１５０４からステップＳ５０２へ分岐したときの条件（Dh≧Ｎ）から、明白である。

【０１２３】同様に、ステップＳ５０３、Ｓ１４０４、Ｓ１４０５によるループ処理を抜け出る時のｉの値の上限はｈである。これはステップＳ１５０４からステップＳ１４０２へ分岐したときの条件（Dh＜Ｎ）から分かる。

【０１２４】本実施形態では、ｋパラメータを決定するまでのループ処理の上限ループ回数を約半分にすることができるので、より高速にｋパラメータを決定することができる。

【０１２５】また、本実施形態の応用として、ｋパラメータを決定するループ処理への分岐数を増やせば、各ループ処理の上限ループ回数を更に減らすことができ、より高速にｋパラメータを決定することができる。

【０１２６】［第９の実施形態］前記第９の実施形態のような初期値の設定方法では、制御変数ｉが増加する場合のループ処理と減少する場合のループ処理の２つを用意しなければならない。そこで、該２つのループ処理を共通化した場合の処理形態を図１６を用いて説明する。

【０１２７】本実施形態では、ステップＳ１６０１にて、ｉの初期値をｈに設定する。ステップ１６０２では、ｉがｈに等しいかどうか判定する。ステップ１６０３では、ｉの初期値を０に再設定すると共に、ｈを－１に設定する。

【０１２８】その他の処理ステップは前記第８の実施形態：図１５の同一番号処理ステップと同じである。

【０１２９】前記第８の実施形態と同様の演算でｈを計算し（ステップＳ１５０２）、ステップＳ１６０１にて、ｉに該ｈを設定する。次に、ステップＳ５０３に移り、$D_h = S_{h-1} - S_h$を計算する。そして、ステップＳ１４０４にて、D_hがＮより大きいかどうか判定する。ここまでの処理は、フローチャートの構造は違ってはいるが、前記第８の実施形態と同じである。

【０１３０】D_h＞Ｎであれば、ｉ＝ｈを初期値として、制御変数ｉが増加する場合のループ処理と同様の処理をステップＳ５０３、Ｓ１４０４、Ｓ１４０５にて行なう。該ループ処理を抜け出る時のｉの値は、少なくとも１回はステップＳ１４０５（ｉ＝ｉ＋１を計算する）を通過するので、前記ｈより大きく、ステップＳ１６０２の判定を通って、ステップＳ１４０７にて、ｋパラメータの値をｉ－１に決定する。

【０１３１】D_h＞Ｎでない場合は、初回からステップＳ１６０２へ分岐する。この時ｉ＝ｈなので、ステップＳ１６０３へ分岐し、ｉの初期値を０に再設定すると共に、ｈを－１に設定する。ｈを－１に設定する理由はｉが取り得ない値を設定することで、２度目のステップ１６０２における判定にひっかからない（ｉ＝ｈとならない）ようにするためである。

【０１３２】ｉの初期他を０に再設定した後の処理は、前記第８の実施形態におけるステップＳ５０３、Ｓ１４０４、Ｓ１４０５によるループ処理と完全に同じで、該ループ処理を抜ける時のｉの値は最大でｆｌｏｏｒ｛ｍ／２｝である。

【０１３３】変数ｈの値はステップＳ１６０３で－１に設定されているので、ステップＳ１６０２にてｉ＝ｈが成り立つことは絶対にないため、ステップＳ１４０７へ分岐し、ｋパラメータの値はｉ－１に決定される。

【０１３４】前記第８の実施形態に対して、処理を共通化したことによるオーバーヘッドは、ｉの初期値を０に再設定する際に必要なステップＳ１６０２の判定処理が初期化時に１／２回（初期化時にこの判定処理を行なう確率は約１／２であるため）と前記ループ処理を抜ける際に１回（この時は該判定処理を避けて通ることができないため）あるが、処理ルーチンのコンパクトさのメリットの方が大きいと思われる。

【０１３５】なお、上記各実施形態におけるその構成は図１に従うものとしたが、例えば同図の処理をプログラムによっても実現できる。画像を入力するデバイスは、ネットワーク、ビデオカメラ、イメージスキャナ、ストレージデバイス（フロッピー等）で良いわけであるから、本発明は、例えばこれらのいずれかを有するシステムや汎用情報処理装置（パーソナルコンピュータ等）にも適用できる。

【０１３６】従って、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体（または記録媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（または

CPUやMPU）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム(OS)などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【０１３７】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【０１３８】以上説明したように本実施形態によれば、あるパラメータに基づいてシンボルを可変長符号部分と固定長符号部分とで符号化する際、そのパラメータの各々の値に対する可変長符号部分の符号量間の差分値のみを評価して、最適なパラメータを高速に決定できるようになった。

【０１３９】

【発明の効果】以上説明したように本発明によれば、多値画像を簡易かつ高速に可変長符号化することのできるようになる。

【図面の簡単な説明】

【図１】実施形態の装置のブロック構成図である。

【図２】離散ウェーブレット変換を施す様子を示す図である。

【図３】周波数成分（サブブロック）に用いる量子化ステップを示す図である。

【図４】Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化をして得られる符号の一例を示す図である。

【図５】第１の実施形態である処理内容を示すフローチャートである。

【図６】ｋパラメータの各値に対する可変長符号の符号量Spを計算するプログラムの一例を示す図である。

【図７】Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化した符号をビットプレーンで表現した図である。

【図８】最終的に出力されるビットストリームの様子を示す図である。

【図９】第２の実施形態である処理内容を示すフローチャートである。

【図１０】符号量Spと各ビットプレーン中の１の数Bpの両方を計算するプログラムを示す図である。

【図１１】第４の実施形態である処理内容を示すフローチャートである。

【図１２】第５の実施形態である処理内容を示すフローチャートである。

【図１３】第６の実施形態である処理内容を示すフローチャートである。

【図１４】第７の実施形態である処理内容を示すフローチャートである。

【図１５】第８の実施形態である処理内容を示すフローチャートである。

【図１６】第９の実施形態である処理内容を示すフローチャートである。

【符号の説明】

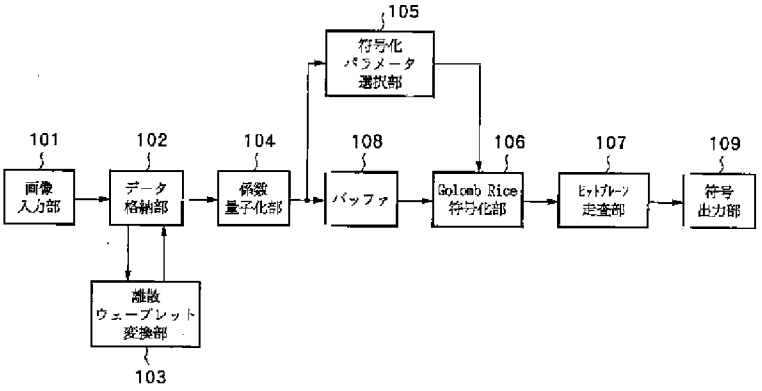
１０１　画像入力部
１０２　データ格納部
１０３　離散ウェーブレット変換部
１０４　係数量子化部
１０５　符号化パラメータ選択部
１０６　Ｇｏｌｏｍｂ　Ｒｉｃｅ符号化部
１０７　ビットプレーン走査部
１０８　バッファ
１０９　符号出力部

【図６】

```
/* サブブロック内のすべての絶対値化した量子化値Ｖに対して
        以下の処理を行なう */

int   t＝Ｖ；
int   p＝0；                    /* 初期化 */
do{
    S[p＋＋]＋＝t；             /* Sp＝Sp＋(Ｖ>>p),p＝p＋1 */
}while(t>>＝1)                  /* (Ｖ>>p)＞0なら処理を継続 */
```

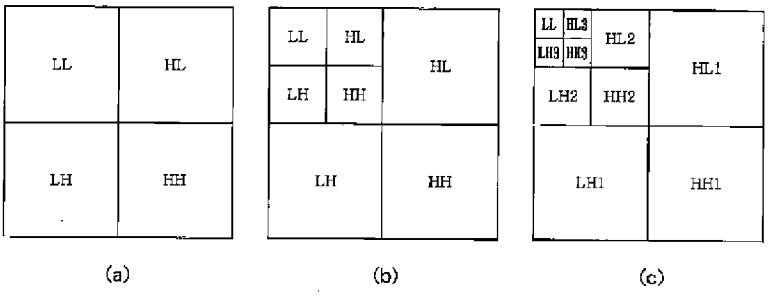
【図1】



【図3】

周波数成分	量子化ステップq
LL	1
HL3	2
LH3	2
HH3	2
HL2	4
LH2	4
HH2	4
HL1	8
LH1	8
HH1	8

【図2】



【図4】

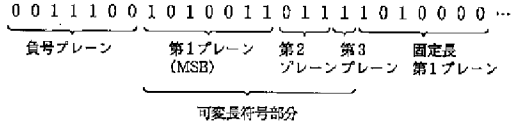
V \ k	0	1	2
0	1	10	100
1	01	11	101
2	001	010	110
3	0001	011	111
4	00001	0010	0100
5	000001	0011	0101
6	0000001	00010	0110
7	00000001	00011	0111

【図7】

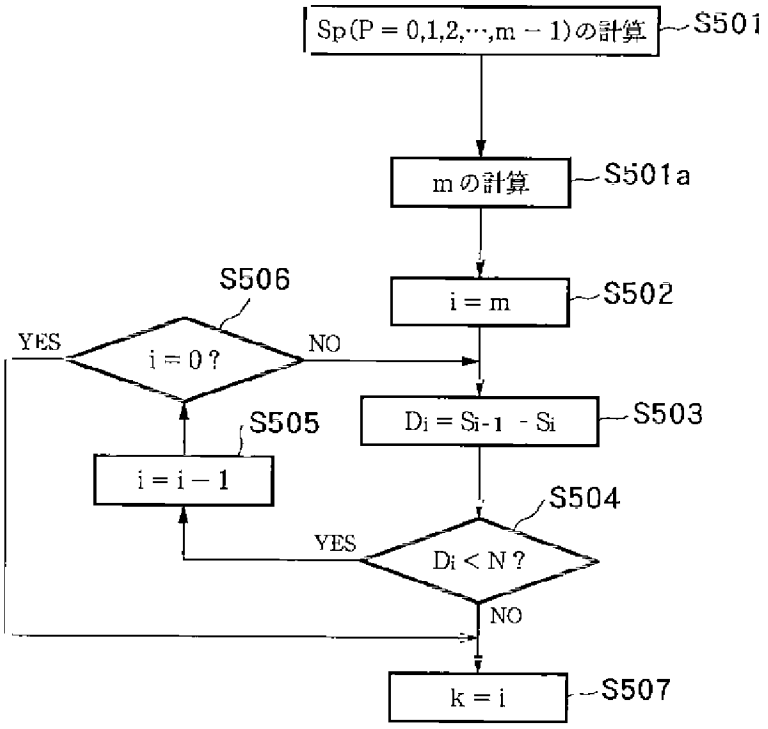
	3	8	−2	-5	4	0	1
負号ビット	0	0	1	1	1	0	0
可変長符号領域	1	0	1	0	0	1	1
		0		1	1		
		1					
固定長符号領域	1	0	1	0	0	0	0
	1	0	0	1	0	0	1

k = 2

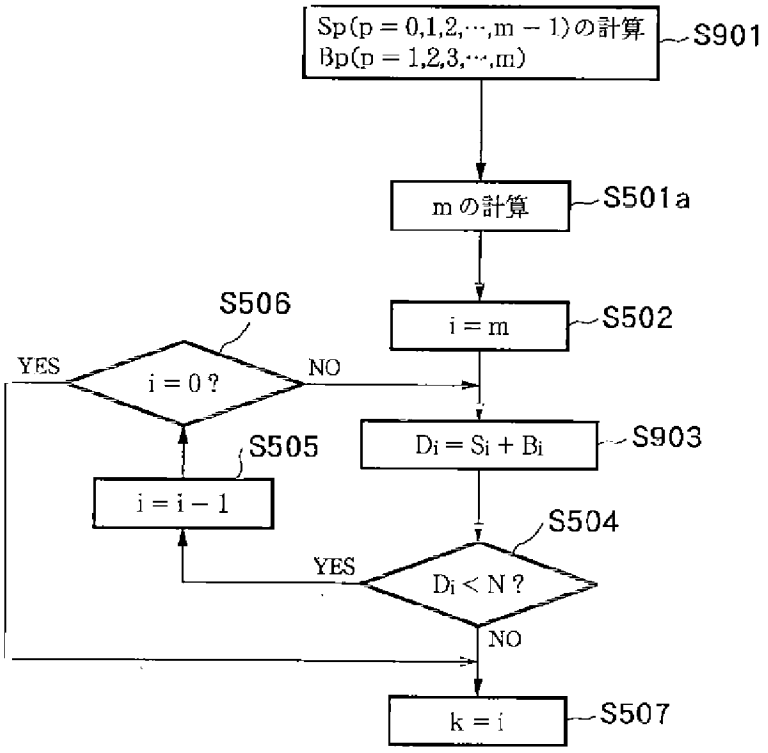
【図8】



【図5】


Sp(P = 0,1,2,…,m − 1)の計算
S501
mの計算
S501a
i = m
S502
S506
i = 0 ?
YES
NO
Di = Si-1 - Si
S503
S505
i = i − 1
S504
YES
Di < N ?
NO
k = i
S507

【図9】

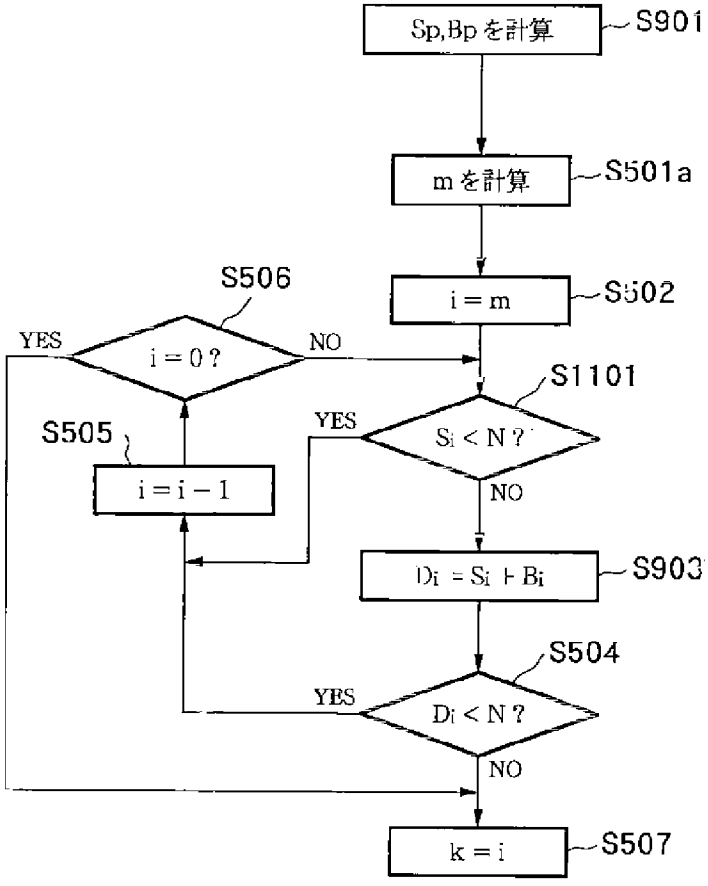

Sp(p = 0,1,2,…,m − 1)の計算
Bp(p = 1,2,3,…,m)
S901
mの計算
S501a
i = m
S502
S506
i = 0 ?
YES
NO
Di = Si + Bi
S903
S505
i = i − 1
S504
YES
Di < N ?
NO
k = i
S507

【図10】

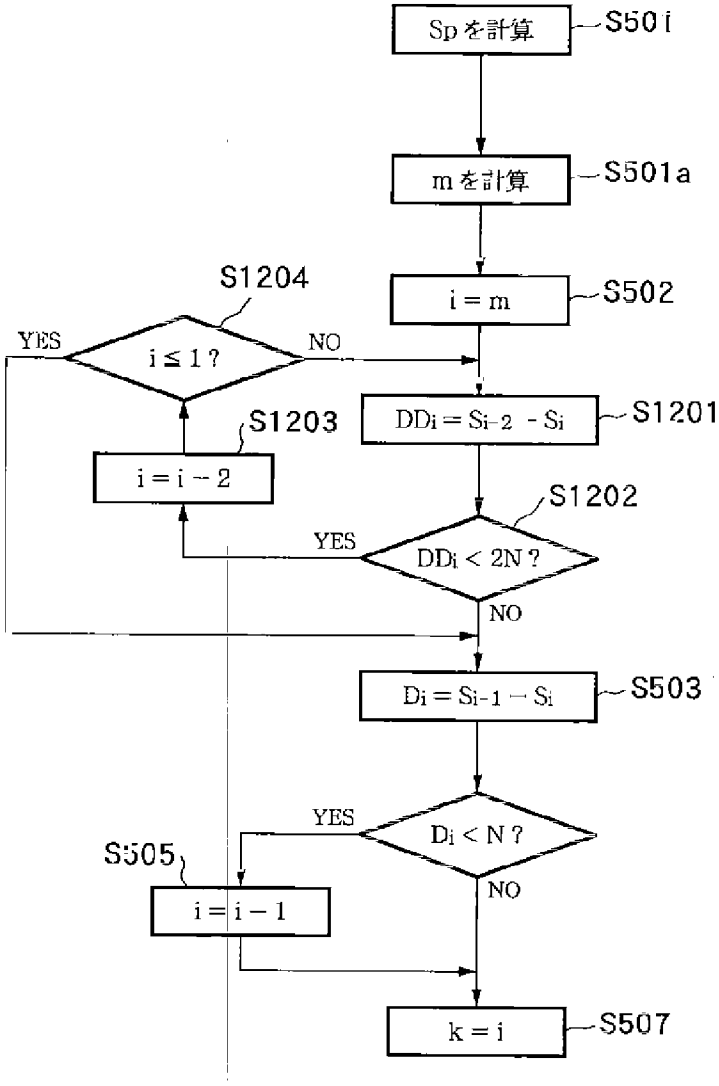
```
/* サブブロック内のすべての絶対値化した量子化値Vに対して
       以下の処理を行なう */

int  t = V ;
int  p = 0 ;                    /* 初期化 */
do{
    S[p++] += t ;               /* Sp = Sp + (V>>p),p = p + 1  */
    B[p] += (t & 1);            /* Bp = Bp + ((V>>p) & 1)      */
}while(t>>=1)                   /* (V>>p) > 0 なら処理を継続    */
```

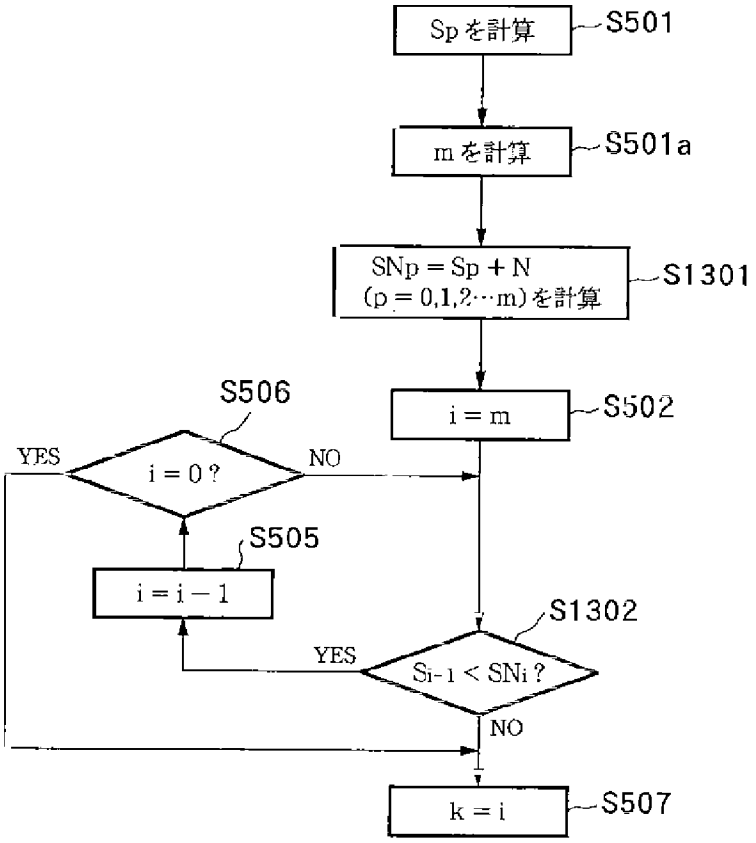
【図11】



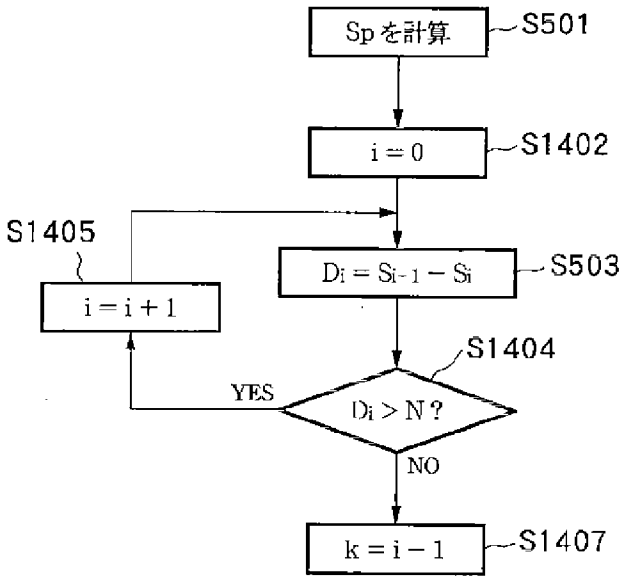
【図12】


Spを計算
S501
mを計算
S501a
i = m
S502
S1204
YES
i ≤ 1 ?
NO
DDi = Si-2 - Si
S1201
S1203
i = i − 2
S1202
YES
DDi < 2N ?
NO
Di = Si-1 − Si
S503
YES
Di < N ?
S505
NO
i = i − 1
k = i
S507

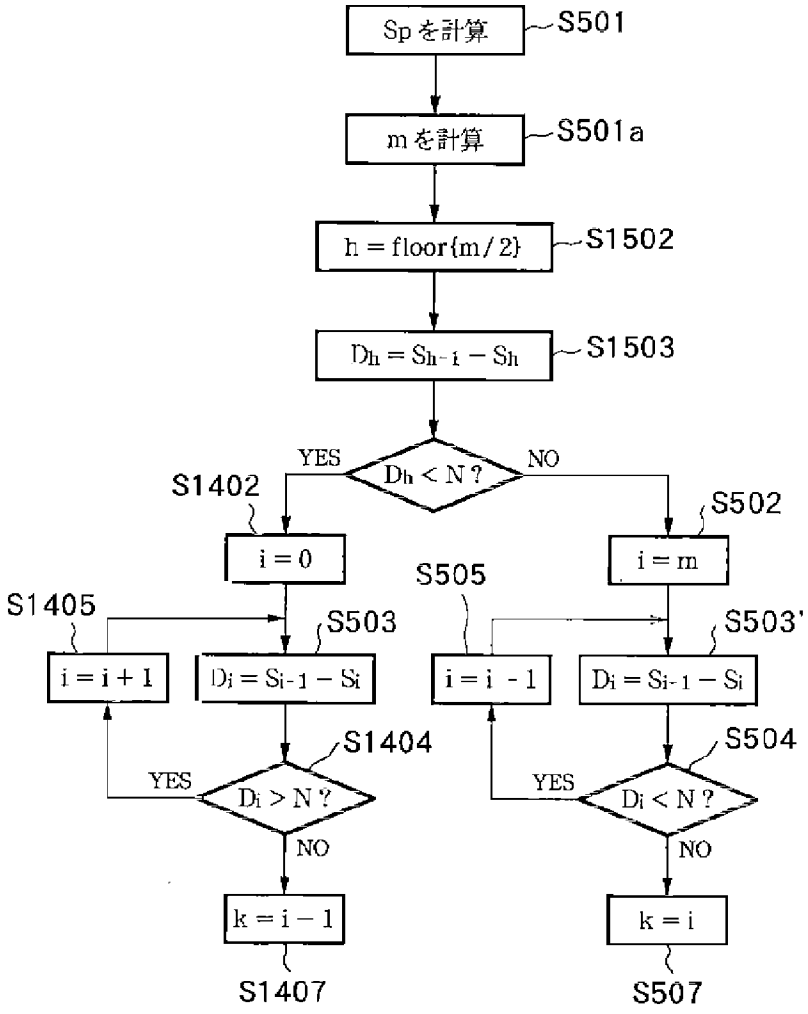
【図13】


Spを計算
S501
mを計算
S501a
SNp＝Sp＋N
(p＝0,1,2…m)を計算
S1301
i＝m
S502
S506
i＝0？
YES
NO
S505
i＝i－1
S1302
Si-1＜SNi？
YES
NO
k＝i
S507

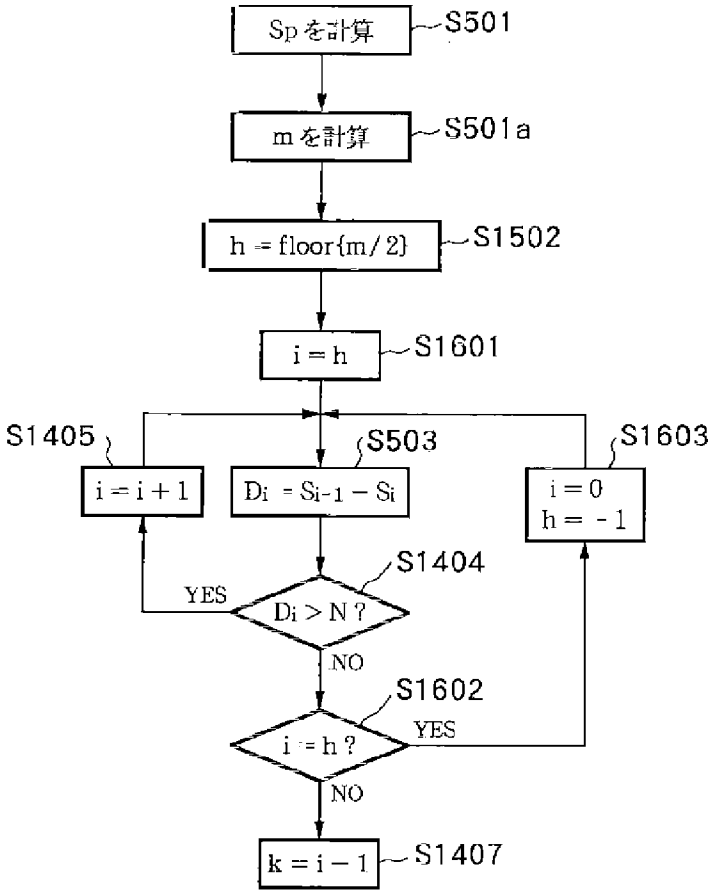
【図14】


Spを計算
S501
i＝0
S1402
S1405
Di＝Si-1－Si
S503
i＝i＋1
S1404
YES
Di＞N？
NO
k＝i－1
S1407

【図15】


Spを計算
S501
mを計算
S501a
h＝floor{m/2}
S1502
Dh＝Sh-i－Sh
S1503
YES
Dh＜N？
NO
S1402
S502
i＝0
i＝m
S505
S1405
S503
S503'
i＝i＋1
Di＝Si-1－Si
i＝i －1
Di＝Si-1－Si
S1404
S504
YES
Di＞N？
YES
Di＜N？
NO
NO
k＝i－1
k＝i
S1407
S507

【図16】



フロントページの続き

Fターム(参考) 5C059 KK11 MA24 MA35 MC15 ME01
ME13 ME17 PP15 PP16 SS13
SS20 SS26 TA48 TB08 TC00
TD05 UA02 UA34
5C078 AA09 BA21 BA53 CA31 CA35
DA01 DA07 DB07 DB19
5J064 AA03 BA09 BA16 BB01 BC02
BC16 BC25 BD01
9A001 EE04 GG17 HH27 JJ71 KK54